C

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-053187
(43)Date of publication of application : 26.02.1999

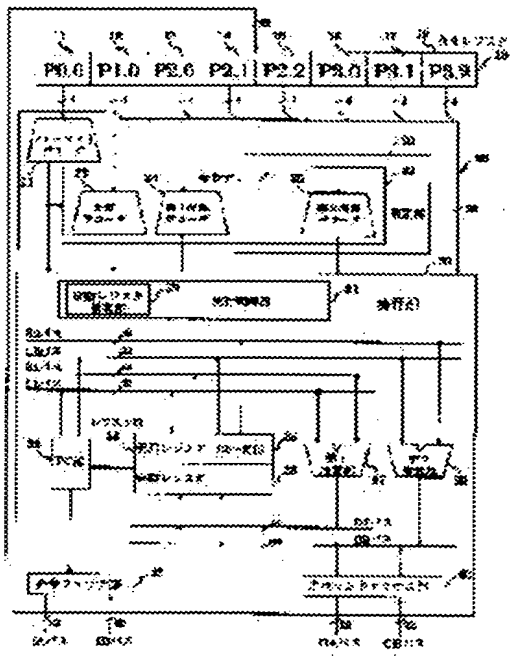
(51)Int.Cl. G06F 9/30
G06F 9/305
G06F 9/38

(21)Application number : 09-204125
(22)Date of filing : 30.07.1997

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD
(72)Inventor : HEIJI TAKEHITO
HIGAKI NOBUO
TANAKA AKIRA
TANAKA TETSUYA
TAKAYAMA SHUICHI
KOTANI KENSUKE
MIYAJI SHINYA

(54) PROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To decrease the number of execution cycles by eliminating an overhead due to separation of operation for accumulating a constant and operation for using it by using even a 1st constant stored in a constant register in combination when a 2nd constant is used as an operand.
SOLUTION: A decoding part 20 decodes an instruction held in an instruction register 10 and outputs the control line corresponding to the decoding result to an execution part 30. When a state wherein a 1st constant is placed in the instruction is decoded, the 1st constant is stored in a constant register 36 and when a state wherein a 2nd constant is placed is decoded, the constant stored in the constant register 36 is read out and combined with the 2nd constant to obtain an operand. Therefore, while a constant as an operand is generated from the constant accumulated in the constant register 36 and the constant specified by operation, the constant is used to perform operation.



LEGAL STATUS

[Date of request for examination] 09.04.2001
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平11－53187

(43)公開日　平成11年(1999)２月26日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	9/30	３５０	Ｇ０６Ｆ	9/30	３５０Ｆ
	9/305			9/38	３７０Ｘ
	9/38	３７０		9/30	３４０Ｆ

審査請求　未請求　請求項の数10　ＯＬ　（全 22 頁）

(21)出願番号　特願平9－204125

(22)出願日　平成９年(1997)７月30日

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　瓶子　岳人
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　檜垣　信夫
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　田中　旭
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(74)代理人　弁理士　滝本　智之　（外１名）

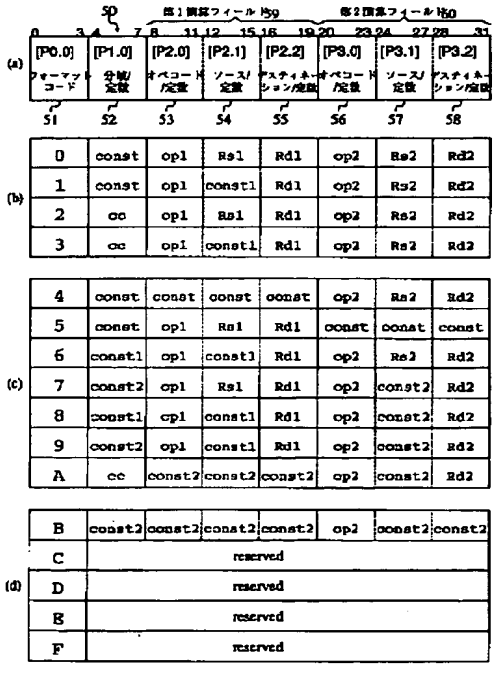
最終頁に続く

(54)【発明の名称】　プロセッサ

(57)【要約】

【課題】　命令を実行するプロセッサにおいて、命令中の無駄領域を後続命令で使用する定数を分割して埋めるために実装される定数レジスタの値の、効率的な蓄積・利用方法により、実行サイクル数の低減を図る。

【解決手段】　上記定数レジスタに蓄積された定数とオペレーションで指定された分割された定数とを連結してオペランドとする定数を生成すると同時に、その定数をオペランドとしてオペレーションを実行することを可能とする。さらに、上記定数レジスタの内容を参照するのみで、定数レジスタの内容を変化させないオペレーションを備える。また、上記定数レジスタの内容の一部のみを変更し、残りの部分の内容を変化させないオペレーションを備える。



1

【特許請求の範囲】

【請求項１】　複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、
定数を格納するための定数レジスタと、
前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、
前記解読手段により前記第１の定数が置かれていると解読された場合に、前記第１の定数を前記定数レジスタに格納する格納手段と、
前記解読手段により前記第２の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項２】　複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、
定数を格納するための定数レジスタと、
前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、
前記解読手段により前記第１の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第１の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第１の定数を新たに前記定数レジスタに格納する格納手段と、
前記解読手段により前記第２の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項３】　前記解読手段は、さらに、前記命令中に所定のオペレーションが置かれていることを解読し、
前記実行手段は、
前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとする第１のオペレーションと、読み出された定数のみをオペランドとする第２のオペレーションとを選択的に実行し、
前記解読手段により前記第２の定数が置かれていると解読された場合には前記第１のオペレーションを実行し、
前記解読手段により前記所定のオペレーションが置かれていると解読された場合には、前記所定のオペレーションを前記第２のオペレーションとして実行することを特徴とする請求項１または２記載のプロセッサ。

【請求項４】　複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、

2

定数を格納するための定数レジスタと、
前記命令中に定数が置かれていることを解読する解読手段と、
前記解読手段により定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記定数を新たに前記定数レジスタに格納する格納手段と、
前記定数レジスタに格納されている定数を読み出し、その定数をオペランドとするオペレーションを実行する実行手段と、
前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、
前記格納手段は、
前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とするプロセッサ。

【請求項５】　前記プロセッサは、さらに、
前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段を備え、
前記格納手段は、
前記状態保持手段により定数が読み出されたことを示されていると、前記第１の定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とする請求項１から請求項３いずれか記載のプロセッサ。

【請求項６】　前記実行手段は、
前記状態保持手段により定数が読み出されたことを示されていても、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行することを特徴とする請求項４または請求項５記載のプロセッサ。

【請求項７】　前記解読手段は、さらに、前記命令中に第３の定数を伴う定数置換オペレーションが置かれていることを解読し、
前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第３の定数とを置換して格納することを特徴とする請求項１から請求項３いずれか記載のプロセッサ。

【請求項８】　前記置換する部分が、前記定数レジスタの最下位ビットを含む部分である請求項７記載のプロセッサ。

【請求項９】　複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、
定数を格納するための定数レジスタと、

前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、

前記解読手段により前記第１の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第１の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第１の定数を新たに前記定数レジスタに格納する格納手段と、

前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段と、

前記解読手段により前記第２の定数が置かれていると解読された場合であって、

前記状態保持手段により定数が読み出されていないことを示されている場合は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行し、

前記状態保持手段により定数が読み出されたことを示されている場合は、前記第２の定数のみをオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項１０】　命令のフォーマットを指定するフォーマットコードが置かれるフォーマットフィールドと、並列実行させるオペレーションを指定する複数のオペレーションフィールドとを含む命令を解読し実行するＶＬＩＷ方式のプロセッサであって、

解読手段は、前記フォーマットコードを参照することにより、少なくとも１つの前記オペレーションフィールドに定数が置かれていることを解読することを特徴とする請求項１から請求項９いずれか記載のプロセッサ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、マイクロプロセッサに関し、特に実行サイクル数の減少を図る技術に関する。

【０００２】

【従来の技術】近年のマイクロプロセッサ応用製品の高機能化及び高速化に伴い、コード効率の高いプログラムを実行することができるマイクロプロセッサ（以下、単に「プロセッサ」という。）が望まれている。つまり、プログラムを構成する各命令中には無駄なコードや未使用の領域が含まれないことが望ましい。

【０００３】ところが、特に、ＶＬＩＷ（Very Long Instruction Word）の如く固定長命令の場合には、命令中にノーオペレーションコード（ｎｏｐコード）等の無駄なコードを置く必要が生じる。ＶＬＩＷは複数のオペレーションフィールドからなり、各オペレーションフィールドにおいてプロセッサが備える複数の演算ユニットに

対応したオペレーションが指定されるが、オペレーションの依存関係等により、常に並列実行可能な複数のオペレーションが存在するとは限らないからである。

【０００４】このようなｎｏｐコードの挿入に伴うコード効率の低下を回避する従来の技術として、例えば、特開平８－１６１１６９に開示されたＶＬＩＷ方式の計算機システムがある。これは、ｎｏｐコードを配置する必要があるオペレーションフィールドに、他の命令で用いられる定数を配置しておき、その配置情報を参照して配置してある定数を他の命令のオペランドとして用いる、というシステムである。

【０００５】しかしながら、上記のシステムでは、無駄領域を埋めることができる定数の大きさが制限されるという問題がある。つまり、オペレーションフィールド長よりも長い定数を配置することができない。そこで、もう一つの従来技術として、特願平９－１５９０４９において出願人は、ｎｏｐコードを配置する必要があるオペレーションフィールドに分割した定数を配置し、その定数を内部の定数レジスタに蓄積していき、他の命令でその定数レジスタを指定することにより蓄積した定数をその命令のオペランドとして用いる、というシステムを提案している。このようにして、後続命令で使用される定数を分割して配置していくことにより、命令中にｎｏｐコードを挿入することによる無駄領域の発生を回避している。

【０００６】

【発明が解決しようとする課題】しかしながら、上記の定数を内部の定数レジスタに蓄積していくシステムでは、定数レジスタに定数を蓄積するオペレーションと、定数レジスタに蓄積された定数を使用するオペレーションが分離しており、定数を使用するオペレーションで定数レジスタを指定する必要があるため、必要以上に実行サイクル数が増大するという問題点を有している。

【０００７】また、上記のシステムでは、定数レジスタに蓄積された定数を一度使用すると、次にその定数と全く同じ値や一部しか異ならない値を蓄積する場合でも、一から定数レジスタのすべてのビットを蓄積する必要があり、実行サイクル数が増大するという問題点を有している。そこで、本発明はかかる問題点に鑑みてなされたものであり、定数レジスタに定数を蓄積するオペレーションと、定数レジスタの定数を使用するオペレーションが分離していることによるオーバーヘッドを解決し、実行サイクル数を低減することを目的とする。

【０００８】さらに、定数レジスタに蓄積された定数を一度しか使用できない点を改善し、定数レジスタの内容の一部又は全部を繰り返して読み出せるようにすることにより、実行サイクル数を低減することを目的とする。

【０００９】

【課題を解決するための手段】上記目的を達成するために請求項１記載のプロセッサは、複数の命令を格納する

記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、前記解読手段により前記第１の定数が置かれていると解読された場合に、前記第１の定数を前記定数レジスタに格納する格納手段と、前記解読手段により前記第２の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【００１０】これによって、第２の定数がオペランドとして使用される際に定数レジスタに格納された第１の定数も併せて使用される。また上記目的を達成するために請求項２記載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、前記解読手段により前記第１の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第１の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第１の定数を新たに前記定数レジスタに格納する格納手段と、前記解読手段により前記第２の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【００１１】これによって、複数の命令に跨って分割配置されていた第１の定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。さらに第２の定数がオペランドとして使用される際に定数レジスタに格納復元された第１の定数も併せて使用される。ここで請求項３記載のプロセッサは、請求項１または２記載のプロセッサにおいて、前記解読手段は、さらに、前記命令中に所定のオペレーションが置かれていることを解読し、前記実行手段は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとする第１のオペレーションと、読み出された定数のみをオペランドとする第２のオペレーションとを選択的に実行し、前記解読手段により前記第２の定数が置かれていると解読された場合には前記第１のオペレーションを実行し、前記解読手段により前記所定のオペレーションが置かれていると解読された場合には、前記所定のオペレーションを前記第２のオペレーションとして実行するとしたものである。

【００１２】また上記目的を達成するために請求項４記載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に定数が置かれていることを解読する解読手段と、前記解読手段により定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記定数を新たに前記定数レジスタに格納する格納手段と、前記定数レジスタに格納されている定数を読み出し、その定数をオペランドとするオペレーションを実行する実行手段と、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とする。

【００１３】これによって、複数の命令に跨って分割配置されていた定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。復元された定数は、読み出された後に次に新たな定数の断片が定数レジスタに格納される前に消去される。ここで請求項５記載のプロセッサは、請求項１から３記載のプロセッサにおいて、さらに、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段を備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記第１の定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去するとしたものである。

【００１４】また請求項６記載のプロセッサは、前記実行手段は、請求項４または５記載のプロセッサにおいて、前記状態保持手段により定数が読み出されたことを示されていても、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行するとしたものである。

【００１５】これによって、定数レジスタに復元格納された定数は繰り返し読み出される。さらに請求項７記載のプロセッサは、請求項１から３記載のプロセッサにおいて、前記解読手段は、さらに、前記命令中に第３の定数を伴う定数置換オペレーションが置かれていることを解読し、前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第３の定数とを置換して格納するとしたものである。

【００１６】これによって、定数レジスタに復元格納された定数は一部が置き換えられつつ繰り返し読み出される。ここで請求項８記載のプロセッサは、請求項７記載のプロセッサにおいて、前記置換する部分が、前記定数

レジスタの最下位ビットを含む部分であるとしたものである。

【００１７】さらに上記目的を達成するために請求項９記載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、前記解読手段により前記第１の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第１の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第１の定数を新たに前記定数レジスタに格納する格納手段と、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段と、前記解読手段により前記第２の定数が置かれていると解読された場合であって、前記状態保持手段により定数が読み出されていないことを示されている場合は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行し、前記状態保持手段により定数が読み出されたことを示されている場合は、前記第２の定数のみをオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【００１８】これによって、複数の命令に跨って分割配置されていた第１の定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。第１の定数を初めて読み出す時には第２の定数も併せて使用される。さらに上記した請求項１から９記載の各プロセッサは、命令のフォーマットを指定するフォーマットコードが置かれるフォーマットフィールドと、並列実行させるオペレーションを指定する複数のオペレーションフィールドとを含む命令を解読し実行するＶＬＩＷ方式のプロセッサであって、前記解読手段は、前記フォーマットコードを参照することにより、少なくとも１つの前記オペレーションフィールドに定数が置かれていることを解読するとすることができる。

【００１９】

【発明の実施の形態】以下、本発明に係るプロセッサの実施の形態について、図面を用いて詳細に説明する。なお、本明細書では、「命令」とは本プロセッサが同時並列に解読し実行するコード全体を意味し、「オペレーション」とは本プロセッサが並列に実行できる数値演算、論理演算、転送、分岐等の処理単位又はその処理単位を指定するためのコードを意味する。

（命令フォーマット）まず、本プロセッサが解読実行する命令の構造について説明する。

【００２０】本プロセッサは、ＶＬＩＷアーキテクチャを採るプロセッサ（以下、「ＶＬＩＷプロセッサ」という。）であり、３２ビット固定長の命令を解読実行する。図１（ａ）は、本プロセッサが実行する命令５０のフィールド構成を示す図である。図１（ｂ）～図１（ｄ）は１６種類の命令フォーマットを示す図であり、そのうち、図１（ｂ）は３オペレーション、図１（ｃ）は２オペレーション、図１（ｄ）は１オペレーションを同時に指定できる命令フォーマットである。

【００２１】この命令５０は、３２ビット固定長であり、４ビットずつに区切られた８個のフィールド（上位よりＰ０．０フィールド５１、Ｐ１．０フィールド５２、…、Ｐ３．２フィールド５８）からなる。なお、Ｐ２．０フィールド５３～Ｐ２．２フィールド５５のグループをまとめて第１演算フィールド５９と呼び、Ｐ３．０フィールド５６～Ｐ３．２フィールド５８のグループをまとめて第２演算フィールド６０と呼ぶ。

【００２２】図１（ｂ）～図１（ｄ）において、“ｃｏｎｓｔ”は定数であり、これが用いられるオペレーションの種類によっては即値、絶対番地、ディスプレースメント等の数値定数や文字定数を意味する。“ｏｐ”はオペレーションの種類を指定するオペコードを、“Ｒｓ”はソースオペランドとなるレジスタを、“Ｒｄ”はデスティネーションオペランドとなるレジスタを、“ｃｃ”は本プロセッサが備える専用の３２ビットレジスタ（図３に示される定数レジスタ３６）の格納値を分岐先の絶対番地又は相対番地（ディスプレースメント）とする分岐オペレーションを指定するオペコードを意味する。

【００２３】各命令フォーマット毎の実際の動作については、後で詳細な説明を加える。図２は、図１で用いられている３種類のオペコード“ｃｃ”、“ｏｐ１”及び“ｏｐ２”それぞれによって指定される具体的なオペレーションを説明する図である。４ビットのオペコード“ｃｃ”は、１６種類の分岐オペレーションの中の一つを指定する。１つの分岐オペレーションは、分岐条件と分岐形式によって特定される。分岐条件には、等しい（“ｅｑ”）、等しくない（“ｎｅｑ”）、より大きい（“ｇｔ”）等がある。分岐形式には、上記定数レジスタ３６の格納値を分岐先の絶対番地として分岐する形式（ニーモニック表示において“ｉ”が添付されていないもの）と相対番地として分岐する形式（ニーモニック表示において“ｉ”が添付されているもの）とがある。例えば、オペコード“ｅｑ”は、直前の比較結果が等しい場合に絶対番地指定による分岐を行なうオペレーションを意味し、オペコード“ｅｑｉ”は、直前の比較結果が等しい場合に相対番地指定による分岐を行なうオペレーションを意味する。

【００２４】４ビットのオペコード“ｏｐ１”は、“ａｄｄ”（加算）、“ｓｕｂ”（減算）、“ｍｕｌ”（乗算）、“ａｎｄ”（論理積）、“ｏｒ”（論理和）等の算術論理演算に属するオペレーションの一つを指定する場合と、“ｍｏｖ”（ワード（３２ビット）データの転

送）、“ｍｏｖｈ”（ハーフワードデータの転送）、“ｍｏｖｂ”（バイトデータの転送）等のレジスタ間転送に属するオペレーションの一つを指定する場合とがある。また、転送オペレーションの一つとして、定数レジスタの下位４ビットの内容を置換する“ｓｅｔ１４”オペレーションが用意されている。

【００２５】４ビットのオペコード“ｏｐ２”は、上記オペコード“ｏｐ１”と同様の算術論理演算及びレジスタ間転送に加えて、“ｌｄ”（メモリからレジスタへの１ワードデータのロード）、“ｓｔ”（レジスタからメモリへのワードデータのストア）等のレジスタ・メモリ間転送に属するオペレーションの一つを指定する場合がある。

【００２６】第１演算フィールド５９には、本プロセッサと外部（メモリ）とのデータの転送を伴わないオペレーション（算術論理演算、レジスタ間転送）を指定するためのオペコードとオペランド（ソース及びデスティネーション）との組又は定数が置かれる。第２演算フィールド６０には、上記第１演算フィールド５９の場合に加えて、本プロセッサと外部（メモリ）とのデータの転送を伴うオペレーション（レジスタ・メモリ間転送）を指定するためのオペコードとオペランドとの組が置かれることもある。

【００２７】なお、以上のようなオペレーションの種類の各フィールドへの割り当ては、ノイマン型の本プロセッサにおいては２つ以上の分岐オペレーションを同時に実行する必要がないこと、本プロセッサと外部（メモリ）とのオペランドの入出力ポート（図３におけるオペランドアクセス部４０）を１つに限定していること等に基づく。

（プロセッサのハードウェア構成）次に、本プロセッサのハードウェア構成を説明する。

【００２８】図３は、本発明に係るプロセッサのハードウェア構成を示すブロック図である。本プロセッサは、上述したように、最大３つのオペレーションを並列実行するＶＬＩＷプロセッサであり、大きく分けて、命令レジスタ１０、解読部２０及び実行部３０から構成される。

【００２９】命令レジスタ１０は、命令フェッチ部３９から送られてきた１個の命令を保持する３２ビットのレジスタである。解読部２０は、命令レジスタ１０に保持された命令を解読し、その解読結果に応じた制御線を実行部３０に出力するものであり、大きく分けて、フォーマットデコーダ２１と命令デコーダ２２とからなる。

【００３０】命令デコーダ２２はさらに、Ｐ１．０フィールド１２に保持されたオペコード“ｃｃ”を解読しその結果に基づいてＰＣ部３３を制御する分岐デコーダ２３と、Ｐ２．０フィールド１３に保持されたオペコードを解読しその結果に基づいて第１演算部３７を制御する第１演算デコーダ２４と、Ｐ３．０フィールド１６に保持されたオペコードを解読しその結果に基づいて第２演算部３８及びオペランドアクセス部４０を制御する第２演算デコーダ２５とからなる。

【００３１】フォーマットデコーダ２１は、Ｐ０．０フィールド１１に保持された４ビットのフォーマットコードをデコードすることによって命令レジスタ１０に保持された命令のフォーマットが図１（ｂ）～図１（ｄ）に示された１６種類のうちのいずれであるかを特定し、その結果に応じて分岐デコーダ２３、第１演算デコーダ２４及び第２演算デコーダ２５による解読動作を許可又は禁止したり、実行部３０の定数レジスタ制御部３２を動作させたりする。

【００３２】なお、上記デコーダ２１、２３～２５は、基本的には１サイクルに１つのオペレーションを解読し、実行部３０に制御信号を与える。また、命令レジスタ１０と実行部３０を接続する２６ビットの定数信号線２６は、命令レジスタ１０に置かれた定数やオペランドを実行部３０に転送するためのバスである。実行部３０は、解読部２０での解読結果に基づいて、最大３つのオペレーションを並列実行する回路ユニットであり、実行制御部３１、ＰＣ部３３、レジスタ群３４、第１演算部３７、第２演算部３８、命令フェッチ部３９及びオペランドアクセス部４０からなる。なお、この実行部３０のうち定数レジスタ制御部３２、ＰＣ部３３及び定数レジスタ３６については、別の図面においてさらに詳細な構成を示している。

【００３３】実行制御部３１は、解読部２０での実行結果に基づいて実行部３０の各構成要素３３～４０を制御する制御回路や配線の総称であり、通常のプロセッサが備える構成要素（タイミング制御、動作許可禁止制御、ステータス管理、割り込み制御等の回路）の他に本プロセッサに特有の定数レジスタ制御部３２を有する。定数レジスタ制御部３２は、フォーマットデコーダ２１からの指示に基づいて命令レジスタ１０に保持された４ビット又は１６ビットの定数（ｃｏｎｓｔ）を定数レジスタ３６に格納する制御を行なう。

【００３４】ＰＣ（プログラムカウンタ）部３３は、分岐デコーダ２３による制御の下で、次に解読実行すべき命令が置かれている図示されていない外部メモリ上のアドレスを命令フェッチ部３９に出力する。命令フェッチ部３９は、３２ビットのＩＡ（インストラクションアドレス）バス及び３２ビットのＩＤ（インストラクションデータ）バスを通じて図示されていない外部メモリから命令ブロックをフェッチし、内部の命令キャッシュに保持すると共に、ＰＣ部３３から出力されたアドレスに相当する命令を命令レジスタ１０に供給する。

【００３５】レジスタ群３４は、１６個の３２ビット汎用レジスタ３５と１個の３２ビット定数レジスタ３６から構成される。これら１７個のレジスタ３５、３６に格納された値は、第１演算デコーダ２４及び第２演算デコ

ーダ２５での解読結果に基づいて、第１演算部３７及び第２演算部３８に転送され、ここで演算が施され、又は、ここを単に通過した後に、レジスタ群３４又はオペランドアクセス部４０に送られる。なお、定数レジスタ３６に格納された値は、第１演算部３７及び第２演算部３８での演算に用いられる他に、ＰＣ部３３にも転送され、ここで分岐先となる有効アドレスを生成するために用いられる。

【００３６】第１演算部３７は、２個の３２ビットデータに対して算術論理演算を行なうＡＬＵと乗算を行う乗算器とを内部に有し、第１演算デコーダ２４による制御の下で２種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。第２演算部３８も、第１演算部３７と同様に、２個の３２ビットデータに対して算術論理演算を行なうＡＬＵと乗算を行う乗算器とを内部に有し、第２演算デコーダ２５による制御の下で２種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。

【００３７】オペランドアクセス部４０は、第２演算デコーダによる制御の下でレジスタ群３４と図示されていない外部メモリとの間でオペランドの転送を行なう回路であり、そのオペランドやオペランドアドレスを保持するバッファを内部に有する。具体的には、例えば、命令レジスタ１０のＰ３．１フィールド１６にオペコード“１ｄ”が置かれていた場合には、外部メモリに置かれていた１ワードのデータがオペランドアクセス部４０を経てレジスタ群３４のいずれかのレジスタにロードされ、また、オペコード“ｓｔ”が置かれていた場合には、レジスタ群３４のいずれかのレジスタの格納値が外部メモリにストアされる。

【００３８】上記ＰＣ部３３、レジスタ群３４、第１演算部３７、第２演算部３８及びオペランドアクセス部４０は、図示されるように、内部バス（Ｌ１バス、Ｒ１バス、Ｌ２バス、Ｒ２バス、Ｄ１バス、Ｄ２バス）で接続されている。なお、Ｌ１バス及びＲ１バスはそれぞれ第１演算部３７の２つの入力ポートに、Ｌ２バス及びＲ２バスはそれぞれ第２演算部３８の２つの入力ポートに、Ｄ１バス及びＤ２バスはそれぞれ第１演算部３７及び第２演算部３８の出力ポートに接続されている。

（定数レジスタ３６及びその周辺回路の詳細な構成）次に、定数レジスタ３６及びその周辺回路について詳細に説明する。

【００３９】図４は、定数レジスタ３６及びその周辺回路の詳細な構成と接続関係を示すブロック図である。なお、図中の固定値（“０”）２７は、定数“０”を示す４本の信号線の固定的な配線を意味する。定数レジスタ制御部３２は、５個の３入力セレクタ３２ａ～３２ｅと３個の４入力セレクタ３２ｆ～３２ｈとからなり、定数レジスタ３６は、８個の４ビット幅レジスタ３６ａ～３６ｈからなる。なお、各入出力データは並列４ビットである。

【００４０】読み出しフラグ記憶部２８は、定数レジスタ３６をゼロクリアすべきかどうかを調べるために用いられる。図５は、読み出しフラグ記憶部２８の値の変化を示す状態遷移図である。読み出しフラグ記憶部２８の値は、定数レジスタ３６の格納値が読み出され、かつ定数レジスタ３６への定数の格納が行なわれなかった場合に“１”にセットされる。読み出しフラグ記憶部２８の値が“１”の状態で、定数を格納するオペレーションが指定された場合、定数を設定する前に定数レジスタ制御部３２の制御により定数レジスタ３６の内容はゼロクリアされる。その後、読み出しフラグ記憶部２８の値は“０”にセットされる。

【００４１】定数レジスタ制御部３２は、フォーマットデコーダ２１及び命令デコーダ２２からの制御信号に従って上記８個の入力セレクタ３２ａ～３２ｈを制御することで、以下に示す８通りの格納方法のいずれかの方法により、命令レジスタ１０に保持された定数を定数レジスタ３６に格納させる。図６（ａ）～図６（ｈ）は、その８通りの格納方法を説明する図である。

【００４２】図６（ａ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“０”又は“１”であると解読され、読み出しフラグ記憶部２８の値が“０”である場合の格納方法を示す。これは、Ｐ１．０フィールド１２に置かれた４ビットの定数のみを定数レジスタ３６に格納する場合に相当する。具体的には、定数レジスタ３６に保持されたデータを４ビット単位で上位にシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２に保持された４ビットの定数を定数レジスタ３６の最下位の４ビットレジスタ３６ｈに格納する。

【００４３】図６（ｂ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“０”又は“１”であると解読され、読み出しフラグ記憶部２８の値が“１”である場合の格納方法を示す。これは、定数レジスタ３６の内容をゼロクリアしてから、Ｐ１．０フィールド１２に置かれた４ビットの定数のみを定数レジスタ３６に格納する場合に相当する。

【００４４】具体的には、命令レジスタ１０のＰ１．０フィールド１２に保持された４ビットの定数を定数レジスタ３６の最下位の４ビットレジスタ３６ｈに格納すると同時に、定数レジスタ３６の残りのビット３６ａ～３６ｇに“０”を格納する。図６（ｃ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“４”であると解読され、読み出しフラグ記憶部２８の値が“０”である場合の格納方法を示す。これは、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。

【００４５】具体的には、定数レジスタ３６の下位１６

ビット３６ｅ～３６ｈに保持されたデータを上位１６ビット３６ａ～３６ｄにシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。図６（ｄ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“４”であると解読され、読み出しフラグ記憶部２８の値が“１”である場合の格納方法を示す。これは、定数レジスタ３６の内容をゼロクリアしてから、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。

【００４６】具体的には、命令レジスタ１０のＰ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納すると同時に、残りのビット３６ａ～３６ｄに“０”を格納する。図６（ｅ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“５”であると解読され、読み出しフラグ記憶部２８の値が“０”である場合の格納方法を示す。これは、Ｐ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。

【００４７】具体的には、定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに保持されたデータを上位１６ビット３６ａ～３６ｄにシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。

【００４８】図６（ｆ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“５”であると解読され、読み出しフラグ記憶部２８の値が“１”である場合の格納方法を示す。これは、定数レジスタ３６の内容をゼロクリアしてから、Ｐ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。

【００４９】具体的には、命令レジスタ１０のＰ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納すると同時に、残りのビット３６ａ～３６ｄに“０”を格納する。図６（ｇ）は、第１演算デコーダ２４でデコードした結果、オペレーションが“ｓｅｔ１４”であると解読された場合の格納方法を示す。これは、現在定数レジスタ３６に格納されている値の下位４ビットのみを変更する場合に相当する。

【００５０】具体的には、定数レジスタ３６に既に格納されているデータのシフトは行なわず、Ｐ２．１フィールドに保持された４ビットの定数を定数レジスタ３６の下位４ビット３６ｈに格納する。図６（ｈ）は、第２演算デコーダ２５でデコードした結果、オペレーションが“ｓｅｔ１４”であると解読された場合の格納方法を示す。これは、現在定数レジスタ３６に格納されている値の下位４ビットのみを変更する場合に相当する。

【００５１】具体的には、定数レジスタ３６に既に格納されているデータのシフトは行なわず、Ｐ３．１フィールドに保持された４ビットの定数を定数レジスタ３６の下位４ビット３６ｈに格納する。以上のように、命令レジスタ１０のＰ０．０フィールド１１の値が“０”、“１”、“４”、“５”であり、読み出しフラグ２８の値が“０”の場合には、定数レジスタ３６に既に格納された定数をシフトさせながら新たな定数が定数レジスタ３６に格納される。そして、命令レジスタ１０のＰ０．０フィールド１１の値が“０”、“１”、“４”、“５”であり、読み出しフラグ２８の値が“１”の場合には、定数レジスタ３６をゼロクリアしてから新たな定数が定数レジスタ３６に格納される。また、“ｓｅｔ１４”オペレーションが指定された場合には、定数レジスタ３６の下位４ビットのみを変更し、上位ビットの格納値はそのまま残すことができる。なお、“ｓｅｔ１４”オペレーションは、すべてのオペコード“ｏｐ１”又は“ｏｐ２”に配置することができ、その場合の“Ｒｄ１”又は“Ｒｄ２”の内容は無視される。

【００５２】“ｓｅｔ１４”のように、定数レジスタ３６の下位ビットのみを変更するオペレーションを用意することにより、下位ビットのみが異なるいくつかの定数を使用したい場合に、定数全体を何度も設定しなおす必要がなくなり、定数レジスタ３６の上位部分を使い回しすることができるようになる。図７（ａ）～図７（ｆ）は、定数レジスタ３６及び読み出しフラグ記憶部２８の値の変化を示す図である。

【００５３】図７（ａ）は、定数レジスタ３６に格納されていた定数“０ｘ８７６５４３２１”が読み出された直後におけるそれらの内容を示し、図７（ｂ）～図７（ｅ）は、その後に４ビットの定数“０ｘ８”、“０ｘ７”、“０ｘ６”、“０ｘ５”が順次に格納された直後におけるそれらの内容を示し、図７（ｆ）は、図７（ｅ）における定数“０ｘ００００８７６５”が読み出された直後におけるそれらの内容を示す。

【００５４】このように、読み出しフラグ記憶部２８によってゼロクリアの必要性の有無を管理することで、複数の命令に跨って分割配置されていた定数は、定数レジスタ３６に蓄積して格納され、ゼロ拡張された元の定数として復元される。ここで、ゼロ拡張とは、ある数値の有効桁数が一定の桁数に満たない場合に、その有効桁より上位の桁全てをゼロで埋める処理をいう。

【００５５】また、読み出しフラグ記憶部２８を用意す

ることにより、新しい定数を格納する際にゼロクリアの必要性の有無を判定することができるようになり、定数レジスタ３６の値を使用した時点で定数レジスタ３６をゼロクリアしておく必要がなくなっている。これによって、以下に示すような定数レジスタ３６の格納値の全体の再利用が可能となっている。

【００５６】ここで、再び図１に示した各命令フォーマットについて、読み出しフラグ記憶部２８の値に応じた本プロセッサの動作を説明する。なお、本プロセッサでは、初期状態として読み出しフラグ記憶部２８の値は“１”にセットされている。フォーマットコード（Ｐ０．０フィールド１１の格納値）“０”について：

（１）読み出しフラグ記憶部２８の値が“１”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００５７】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、Ｐ１．０フィールド５２に指定された“ｃｏｎｓｔ”を定数レジスタ３６の３６ｈに格納し、３６ａ～３６ｇをゼロクリアする。そして、読み出しフラグ記憶部２８の値を“０”にする。

（２）読み出しフラグ記憶部２８の値が“０”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００５８】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、定数レジスタ３６の３６ｂ～３６ｈを３６ａ～３６ｇに４ビット左シフトしてから、Ｐ１．０フィールド５２に指定された“ｃｏｎｓｔ”を定数レジスタ３６の３６ｈに格納する。読み出しフラグ記憶部２８の値は“０”のままである。

【００５９】フォーマットコード“１”について：

（１）読み出しフラグ記憶部２８の値が“１”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”である。

【００６０】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、Ｐ１．０フィールド５２に指定された“ｃｏｎｓｔ”を定数レジスタ３６の３６ｈに格納し、３６ａ～３６ｇをゼロクリアする。そして、読み出しフラグ記憶部２８の値を“０”にする。

（２）読み出しフラグ記憶部２８の値が“０”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、定数レジスタ３６の３６ｂ～３６ｈとＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”を連結した３２ビットの定数である。そして、定数レジスタ３６の内容を読み出したので、読み出しフラグ記憶部２８の値を“１”にセットする。

【００６１】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、この時点で読みだしフラグ記憶部２８の値は“１”になっているので、Ｐ１．０フィールド５２に指定された“ｃｏｎｓｔ”を定数レジスタ３６の３６ｈに格納し、３６ａ～３６ｇをゼロクリアする。そして、読み出しフラグ記憶部２８の値を“０”にする。

【００６２】フォーマットコード“２”について：

（１）読み出しフラグ記憶部２８の値が“１”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。

【００６３】次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。読み出しフラグ記憶部２８の値は“１”のままである。

（２）読み出しフラグ記憶部２８の値が“０”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００６４】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。

【００６５】最後に、読み出しフラグ記憶部２８の値を“１”にセットする。フォーマットコード“３”について：

（１）読み出しフラグ記憶部２８の値が“１”のとき

まず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”であ

る。

【００６６】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。

【００６７】読み出しフラグ記憶部２８の値は“１”のままである。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”である。

【００６８】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。

【００６９】最後に、読み出しフラグ記憶部２８の値を“１”にセットする。フォーマットコード“４”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。

【００７０】次に、Ｐ１．０フィールド５２～Ｐ２．２フィールド５５に指定された１６ビットの定数“ｃｏｎｓｔ”を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納し、３６ａ～３６ｄをゼロクリアする。そして、読み出しフラグ記憶部２８の値を“０”にする。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。

【００７１】次に、定数レジスタ３６の下位１６ビット３６ｅ～３６ｈを３６ａ～３６ｄにシフトしてから、Ｐ１．０フィールド５２～Ｐ２．２フィールド５５に指定された１６ビットの定数“ｃｏｎｓｔ”を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。読み出しフラグ記憶部２８の値は“０”のままである。フォーマットコード“５”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００７２】次に、Ｐ１．０フィールド５２とＰ３．０フィールド５６～Ｐ３．２フィールド５８に指定された１６ビットの定数“ｃｏｎｓｔ”を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納し、３６ａ～３６ｄをゼロクリアする。そして、読み出しフラグ記憶部２８の値を“０”にする。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００７３】次に、定数レジスタ３６の下位１６ビット３６ｅ～３６ｈを３６ａ～３６ｄにシフトしてから、Ｐ１．０フィールド５２とＰ３．０フィールド５６～Ｐ３．２フィールド５８に指定された１６ビットの定数“ｃｏｎｓｔ”を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。読み出しフラグ記憶部２８の値は“０”のままである。

【００７４】フォーマットコード“６”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”を連結させた８ビットの定数である。

【００７５】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、定数レジスタ３６の下位２４ビット３６ｃ～３６ｈとＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”を連結した３２ビットの定数である。そして、定数レジスタ３６の内容を読み出したので、読み出しフラグ記憶部２８の値を“１”にセットする。

【００７６】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。ソースオペランドは“Ｒｓ２”、デスティネーションオペランドは“Ｒｄ２”である。フォーマットコード“７”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００７７】それと同時に、第２演算フィールド６０に

指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドはＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ１”を連結させた８ビットの定数である。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。ソースオペランドは“Ｒｓ１”、デスティネーションオペランドは“Ｒｄ１”である。

【００７８】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドは、定数レジスタ３６の下位２４ビット３６ｃ～３６ｈとＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ１”を連結した３２ビットの定数である。そして、定数レジスタ３６の内容を読み出したので、読み出しフラグ記憶部２８の値を“１”にセットする。

【００７９】フォーマットコード“８”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”を連結させた８ビットの定数である。

【００８０】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドはＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”である。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、定数レジスタ３６の下位２４ビットとＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｎｓｔ１”を連結した３２ビットの定数である。そして、定数レジスタ３６の内容を読み出したので、読み出しフラグ記憶部２８の値を“１”にセットする。

【００８１】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドはＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”である。フォーマットコード“９”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドはＰ２．１フィールド５４に指定された４ビットの定数“ｃｏｕｓｔ１”である。

【００８２】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドはＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ２”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”を連結させた８ビットの定数である。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第１演算フィールド５９に指定された“ｏｐ１”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、定数レジスタ３６の下位２８ビット３６ｂ～３６ｈとＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ１”とを連結した３２ビットの定数である。そして、定数レジスタ３６の内容を読み出したので、読み出しフラグ記憶部２８の値を“１”にセットする。

【００８３】それと同時に、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ２”である。ソースオペランドはＰ１．０フィールド５２に指定された４ビットの定数“ｃｏｕｓｔ２”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”を連結させた８ビットの定数である。

【００８４】フォーマットコード“Ａ”について：

（１）読み出しフラグ記憶部２８の値が“１”のときまず、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、Ｐ２．０フィールド５３～Ｐ２．２フィールド５５に指定された１２ビットの定数“ｃｏｕｓｔ２”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”を連結させた１６ビットの定数である。

【００８５】次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。読み出しフラグ記憶部２８の値は“１”のままである。

（２）読み出しフラグ記憶部２８の値が“０”のときまず、第２演算フィールド６０に指定された“ｏｐ２”を実行する。デスティネーションオペランドは“Ｒｄ１”である。ソースオペランドは、Ｐ２．０フィールド５３～Ｐ２．２フィールド５５に指定された１２ビットの定数“ｃｏｕｓｔ２”とＰ３．１フィールド５７に指定された４ビットの定数“ｃｏｕｓｔ２”を連結させた

１６ビットの定数である。

【００８６】次に、Ｐ１．０フィールド５２に指定された“ｃｃ”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、ＰＣ部３３において定数レジスタ３６の内容から分岐先アドレスを求め、ＰＣ（プログラムカウンタ）に格納する。最後に、読み出しフラグ記憶部２８の値を“１”にセットする。

【００８７】フォーマットコード“Ｂ”について：１オペレーションのみを実行する命令であり、読み出しフラグ記憶部２８の値にかかわらず、定数レジスタ３６の内容の参照は行なわない。オペランドは、Ｐ１．０フィールド５２～Ｐ２．２フィールド５５に指定された１６ビットの定数“ｃｏｕｓｔ２”とＰ３．１フィールド５７～Ｐ３．２フィールド５８に指定された８ビットの定数“ｃｏｕｓｔ２”を連結させた２４ビットの定数である。

【００８８】フォーマットコード“Ｃ”～フォーマットコード“Ｆ”については、“ｒｅｓｅｒｖｅｄ”として、拡張命令などのために取ってある。本プロセッサでは、１個の命令に最大３つのオペレーションを指定することができるが、その場合には、図１（ｂ）に示された３オペレーション用の命令フォーマットから判るように、それら３つのオペレーションの種類は次のいずれかの組み合わせになる。

（１）４ビットの定数を定数レジスタ３６にセットするオペレーションと２個の汎用オペレーション（フォーマットコードが“０”、“１”の場合）

（２）定数レジスタ３６にセットされた値を絶対番地又は相対番地として分岐するオペレーションと２個の汎用オペレーション（フォーマットコードが“２”、“３”の場合）

このように、本プロセッサの命令は、わずか３２ビット長でありながら最大３つのオペレーションを同時に指定することができるコード効率の高いフィールド構成を有している。また、定数レジスタ３６の格納値を読み出すと同時に、読み出した定数とオペランドフィールドに指定された定数を連結した定数をオペランドとしてオペレーションを実行することができることが特徴である。

（ＰＣ部３３の詳細な構成）次に、ＰＣ部３３の詳細な構成を説明する。

【００８９】図８は、ＰＣ部３３の詳細な構成を示すブロック図である。ＰＣ部３３は、定数“４”を示す固定的な配線である固定値（“４”）３３ａ、２入力セレクタ３３ｂ、加算器３３ｃ、次に解読実行すべき命令のアドレスを保持するＰＣ３３ｄ及び４入力セレクタ３３ｅから構成される。このＰＣ部３３では、解読部２０からの制御信号に従ってセレクタ３３ｂ、３３ｅが動作することにより、以下の３通りの値のいずれかが有効アドレスとしてセレクタ３３ｅから命令フェッチ部３９に出力される。

（１）ＰＣ３３ｄの内容に“４”を加算した値
これは、分岐しないで順次に実行する場合、即ち、解読実行された命令に分岐オペレーションが指定されていない場合に相当する。なお、“４”を加算するのは、１つの命令の長さが４バイト（３２ビット）であることによる。

（２）ＰＣ３３ｄの内容に定数レジスタ３６の内容を加算した値
これは、定数レジスタ３６の内容を相対番地として分岐する場合、例えば、Ｐ１．０フィールド１２によって相対番地による分岐が指定されていると分岐デコーダ２３が解読した場合が該当する。

（３）定数レジスタ３６の内容
これは、定数レジスタ３６の内容を絶対番地として分岐する場合、例えば、Ｐ１．０フィールド１２によって絶対番地による分岐が指定されていると分岐デコーダ２３が解読した場合が該当する。

【００９０】以上のように、このＰＣ部３３は、専用の加算器３３ｃを備え、定数レジスタ３６に保持された値を直接用いる構成となっているので、第１演算部３７や第２演算部３８での演算とは独立並行して、定数レジスタ３６の格納値を絶対番地又は相対番地として分岐する実行制御を行なうことができる。

（プロセッサの動作）次に、具体的な命令を解読実行した場合の本プロセッサの動作について説明する。

【００９１】図９は、２４ビットの定数を扱う処理の一例を示すフローチャートである。本図には、レジスタＲ０とＲ１との格納値の差を求め（ステップＳ１００）、その結果に２４ビットの定数“０ｘ８７６５４３”（以下、“０ｘ”を付した数は１６進数を表す）”）を加え（ステップＳ１０１、Ｓ１０２）、最後にレジスタＲ２の格納値をレジスタＲ１に転送しておく（ステップＳ１０３）という処理が示されている。

【００９２】図１０は、図９に示された処理内容を本プロセッサに行なわせるプログラムの例を示す図である。このプログラムは、２個の命令７１、７２から構成されている。１行が１個の命令に相当し、各命令の内容は各フィールドに置かれたニーモニックで表現されている。なお、定数は全て１６進数で表現されている。また、“ｆｍｔｎ（ｎ＝０～Ｆ）”はフォーマットコード“ｎ”を示し、“Ｒｎ（ｎ＝０～１５）”はレジスタ群３４の中の１つのレジスタを示す。

【００９３】図１０を用いて、各命令７１、７２ごとの本プロセッサの動作を説明する。まず、初期状態として、読み出しフラグ記憶部２８の値を“１”にする。

（命令７１）命令７１が命令レジスタ１０にロードされると、フォーマットデコーダ２１は、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ４”）から、この命令はフォーマットコードが“４”の２オペレーション命令であると判断し、以下の２つのオペレーション

が並列実行されるように実行部３０を制御する。
（１）第１のオペレーション
定数レジスタ制御部３２は、内部の８個のセレクタ３２ａ～３２ｈを制御することで、図６（ｃ）に示された格納方法により、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数（０ｘ８７６５）を定数レジスタ３６の下位１６ビットに格納する。定数レジスタ３６の上位１６ビットはゼロクリアされる。そして、読み出しフラグ記憶部２８の値を“０”にする。
（２）第２のオペレーション
第２演算部３８は、汎用レジスタＲ０の内容と汎用レジスタＲ１の内容を入力とし、Ｒ０の内容からＲ１の内容を減算した後に、その結果を再び汎用レジスタＲ０に格納する。
（命令７２）命令７２が命令レジスタ１０にロードされると、フォーマットデコーダ２１は、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ６”）から、この命令はフォーマットコードが“６”の２オペレーション命令であると判断し、以下の２つのオペレーションが並列実行されるように実行部３０を制御する。
（１）第１のオペレーション
Ｐ２．０フィールド１３のオペレーションは、定数をオペランドにとるオペレーションであり、読み出しフラグ記憶部２８の値が“０”なので、定数レジスタ３６の下位２４ビット（０ｘ００８７６５）とＰ１．０フィールド１２に指定された４ビットの定数（０ｘ４）とＰ２．１フィールド１４に指定された４ビットの定数（０ｘ３）を連結した値（０ｘ００８７６５４３）と汎用レジスタＲ０の内容とで加算を行なった後に、その結果を再び汎用レジスタＲ０に格納する。
【００９４】そして、読み出しフラグ記憶部２８の値を“１”にセットする。
（２）第２のオペレーション
第２演算部３８は、汎用レジスタＲ２の値を入力とし、そのまま通過させて、汎用レジスタＲ１に格納する。以上のようにして、暗黙的に定数レジスタ３６の格納値を利用し、定数レジスタ３６とオペレーションで指定された定数を連結した定数をオペランドとして実行することにより、２個の命令によって２４ビットの定数を扱うオペレーションを実行することができる。
（定数レジスタの読み出しと同時に使用することができない場合との比較）次に、本発明に係るプロセッサのように定数レジスタ３６の格納値を暗黙のうちに使用するのではなく、従来の技術の項に示した特願平９－１５９０４９のプロセッサのように、明示的に定数レジスタを指定してオペレーションを行ない、定数レジスタ３６へ定数を格納するオペレーションと定数レジスタ３６の格納値を利用するオペレーションとが分離しているプロセッサにおいて、上記図１０に示されたプログラムと同一の処理を行なったときの処理方法について説明し、本発明に係るプロセッサと比較する。
【００９５】図１１は、上記従来のプロセッサにおいて、図１０と同一内容の処理を行なうプログラムの例を示す。ただし、定数レジスタ名はＲ１５であるとする。図１１と図１０を比較して判るように、上記従来のプロセッサ用のプログラムは、本発明に係るプロセッサ用のものよりも１個の命令だけ多くなっている。図１１においては、１つのオペレーションでは、定数レジスタへの値の設定もしくは定数レジスタの格納値の利用のいずれか一方しか行なうことができないため、２４ビットの定数を設定するのに必要な２個の命令と、その定数を利用する命令とを合わせた３個の命令が必要になっている。その結果、無動作オペレーション“ｎｏｐ”を挿入せざるを得なくなっている。
【００９６】以上のように、本発明に係るプロセッサによれば、定数レジスタに蓄積された定数とオペレーションで指定された定数からオペランドとなる定数を生成すると同時にその定数を用いてオペレーションを実行することができる。これにより、実行サイクル数を減少させることができる。
（定数レジスタ全体の再読み出し）次に、分岐オペレーションを含むプログラムの例を図１２に示す。
【００９７】このプログラムは、３個の命令７６～７８から構成されている。図１２を見て判るように、本プログラムでは、命令７６において図６（ｃ）の格納方法にて１６ビットの定数“０ｘ１２３４”を定数レジスタ３６に設定し、命令７７の“ｅｑｉ”オペレーションにて定数レジスタ３６の内容を参照し、その値（０ｘ１２３４）を相対番地（ディスプレースメント）として分岐オペレーションを実行している。この時点で読み出しフラグ記憶部２８の値は“１”にセットされる。そして、命令７８の“ｇｔｉ”オペレーションにて再び定数レジスタ３６の内容を参照し、その値（０ｘ１２３４）を相対番地として分岐オペレーションを実行している。なお、分岐条件の判定には、直前の“ｃｍｐ”オペレーションの結果が反映される。
【００９８】このように、読み出しフラグ記憶部２８の値が“１”にセットされても、次に定数レジスタ３６に定数を格納するオペレーションが実行されるまでは、定数レジスタ３６の内容は変更されないので、定数レジスタ３６の格納値全体を複数回読み出して利用することができる。これによって、同一の定数を定数レジスタ３６に何度も設定し直す必要がなくなり、実行サイクル数・命令コードサイズを低減することができる。
（定数レジスタの一部を使い回し）次に、図６の定数レジスタ３６への定数の格納方法の説明で現れた“ｓｅｔ１４”オペレーションを含むプログラムの例を図１３に示す。
【００９９】このプログラムは、３個の命令７９～８１

から構成されている。図１３を見て判るように、本プログラムでは、命令７９において図６（ｃ）の格納方法にて１６ビットの定数“０ｘ３２１０”を定数レジスタ３６に設定し、命令８０の“ｅｑｉ”オペレーションにて定数レジスタ３６の内容を参照し、その値（０ｘ３２１０）を相対番地（ディスプレースメント）として分岐オペレーションを実行している。また、命令８０の第二演算フィールド６０の“ｓｅｔ１４”オペレーションによって図６（ｈ）の格納方法にて４ビットの定数“０ｘｃ”を定数レジスタ３６の下位４ビットに格納する。これにより定数レジスタ３６の格納値は“０ｘ３２１ｃ”になる。次に、命令８１の“ｇｔｉ”オペレーションにて定数レジスタ３６の内容を参照し、その値（０ｘ３２１ｃ）を相対番地として分岐オペレーションを実行している。なお、分岐条件の判定には、直前の“ｃｍｐ”オペレーションの結果が反映される。

【０１００】このように、“ｓｅｔ１４”のような定数レジスタ３６の一部のみを変更するオペレーションを用意することにより、一部分のみ異なる定数の設定は、異なる部分のみ設定し直せばよいことになる。これによって、一部分のみ異なる定数を定数レジスタ３６に何度も一から設定し直す必要がなくなり、実行サイクル数・命令コードサイズを低減することができる。

（定数レジスタ３６の周辺回路の変形例）次に、図４に示された定数レジスタ３６の周辺回路についての変形例を示す。上記実施例では、定数レジスタ３６の下位４ビットのみを変更するオペレーションを用意し、上位ビットの格納値を使い回しできることを示したが、下位４ビットだけでなく、任意のビットについて上記のような格納値の一部の再利用が可能である。その一例として、図１４に、定数レジスタ３６の周辺回路の変形例のブロック図を示す。

【０１０１】図４の定数レジスタ制御部３２は定数レジスタ３６をシフトレジスタとして機能させたが、図１４の定数レジスタ制御部９０は定数レジスタ３６を並列入力のレジスタとして機能させている点で異なる。具体的には、定数レジスタ制御部９０とフォーマットデコーダ２１との接続及び定数レジスタ制御部９０の構成要素が図４に示されたものと異なる。

【０１０２】定数レジスタ制御部９０は、格納桁カウンタ９１と８個の８入力セレクタ９０ａ～９０ｈとからなる。格納桁カウンタ９１は、３ビットのカウンタであり、その時点において定数レジスタ３６に蓄積されている定数の有効桁数をニブル（４ビット）単位で示すものである。定数レジスタ制御部９０は、命令レジスタ１０に保持された定数を格納する旨の指示をフォーマットデコーダ２１から受けると、その時点での格納桁カウンタ９１の値を参照することで定数レジスタ３６の適切な位置にその定数を格納し、その後に、格納桁カウンタ９１を更新しておく。

【０１０３】ただし、図４の定数レジスタ制御部３２の場合と異なり、後で実行される命令に置かれた定数が定数レジスタ３６の上位桁に格納される。したがって、例えば３２ビットの定数を１６ビットずつ２回に分けて格納する場合には、先に下位１６ビットを格納することになる。定数レジスタを図１４のように構成し、定数レジスタ３６ａ～３６ｈの任意のレジスタの格納値のみを変更するオペレーション“ｓｅｔ４ａ”、“ｓｅｔ４ｂ”、…、“ｓｅｔ４ｈ”や、定数レジスタ３６の下位８ビットや上位８ビットのみを変更するオペレーション“ｓｅｔ１８”、“ｓｅｔｕ８”などのオペレーション群を用意することにより、定数レジスタ３６の下位のみに限らない一部の格納値のみを変更することが可能となり、残りの格納値を使い回しすることができるようになる。

【０１０４】以上、本発明に係るプロセッサについて、実施形態及び変形例に基づいて説明したが、本発明はこれら実施形態及び変形例に限られないことは勿論である。即ち、

（１）上記実施の形態では、数値定数を扱う例が示されたが、文字定数であってもよいことは言うまでもない。複数の命令に跨って分割配置された文字定数であっても、定数レジスタ３６への複数回の格納によって、桁数の長い元の文字定数が復元されるからである。

（２）また、上記実施の形態では、定数レジスタ３６に定数が格納される時にゼロ拡張される構成になっていたが、格納される時に符号拡張される構成にしてもよいし、また、定数を読み出す際にゼロ拡張／符号拡張を行なう構成にしてもよい。そのためには、定数レジスタ３６の周辺回路に拡張を制御する回路を追加すればよい。

【０１０５】ここで、符号拡張とは、ある数値の有効桁数が一定の桁数に満たない場合に、その数値の最上位ビットを符号ビットとみなし、その符号ビットと同じ論理値で有効桁より上位の桁全てを埋める処理をいう。

（３）また、上記実施の形態では、定数レジスタ３６に格納されていた値の下位部分（上位部分の値を捨てたもの）の下位側に、オペレーションで指定された定数を連結し、それをオペランドとしてオペレーションを実行することを可能としたが、オペレーションで指定された定数を定数レジスタ３６に格納されていた値の下位部分の上位側に連結し、それをオペランドとしてオペレーションを実行することも可能である。さらに、定数レジスタ３６の上位部分の値を捨てるのではなく下位部分の値を捨て、定数レジスタ３６の上位部分の下位側または上位側にオペレーションで指定された定数を連結し、それをオペランドとしてオペレーションを実行することも可能である。

（４）また、上記実施の形態では、図１（ｂ）～図１（ｄ）の命令フォーマットから判るように、１個の命令によって定数レジスタ３６に格納させることができる定

数の桁数は４ビット及び１６ビットのいずれかであったが、本発明はこの桁数に限定されるものではない。例えば、１２ビットや２８ビットの定数を定数レジスタ３６に格納するための命令フォーマットを定義してもよい。そのためには、定数レジスタ３６の周辺回路の接続関係を変更すればよい。

（５）また、上記実施の形態のプロセッサは２個の演算部３７、３８を備えるＶＬＩＷプロセッサであったが、本発明は、１個の演算部のみを備え１個の命令中に１個のオペレーションだけを指定する単一オペレーション命令を実行するＶＬＩＷアーキテクチャを採らないプロセッサに対しても適用することができるのは言うまでもない。

【０１０６】

【発明の効果】以上の説明から明らかなように、本発明のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第１の定数が置かれていることと、前記命令中に第２の定数が置かれていることを解読する解読手段と、前記解読手段により前記第１の定数が置かれていると解読された場合に、前記第１の定数を前記定数レジスタに格納する格納手段と、前記解読手段により前記第２の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第２の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【０１０７】これによって、第２の定数がオペランドとして使用される際に定数レジスタに格納された第１の定数も併せて使用される。その結果、１つのオペレーションで定数の設定とその定数の使用とを同時に行なうこと可能となり、実行サイクル数を削減することができる。また本発明のプロセッサは、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とするプロセッサにおいて、前記実行手段は、前記状態保持手段により定数が読み出されたことを示されていても、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行するとしたものである。

【０１０８】これによって、定数レジスタに復元格納された定数は繰り返し読み出される。その結果、定数レジスタの格納値の全体を複数回使用することができるようになり、全く同じ定数を繰り返し設定する必要がなくなるので、実行サイクル数・命令コードサイズを低減することができる。また本発明のプロセッサは、前記解読手段は、さらに、前記命令中に第３の定数を伴う定数置換オペレーションが置かれていることを解読し、前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第３の定数とを置換して格納するとしたものである。

【０１０９】これによって、定数レジスタに復元格納された定数は一部が置き換えられつつ繰り返し読み出される。その結果、定数レジスタの格納値の一部を複数回使用することができるようになり、一部分のみ異なる定数を繰り返し最初から設定する必要がなくなるので、実行サイクル数・命令コードサイズを低減することができる。

【図面の簡単な説明】

【図１】（ａ）本発明に係るプロセッサが実行する命令のフィールド構成を示す図

（ｂ）３オペレーションを示す図

（ｃ）２オペレーションを示す図

（ｄ）１オペレーションを同時に指定できる命令フォーマットを示す図

【図２】図１で用いられている３種類のオペコード“ｃｃ”、“ｏｐ１”及び“ｏｐ２”それぞれによって指定される具体的なオペレーションを説明する図

【図３】同プロセッサのハードウェア構成を示すブロック図

【図４】同プロセッサの定数レジスタ３６及びその周辺回路の詳細な構成を示すブロック図

【図５】図４に示された読み出しフラグ記憶部２８の値の変化を示す遷移図

【図６】図４に示された定数レジスタ制御部３２による定数の格納方法を示す図

【図７】図４に示された定数レジスタ３６及び読み出しフラグ記憶部２８の値の変化を示す図

【図８】同プロセッサのＰＣ部３３の詳細な構成を示すブロック図

【図９】２４ビットの定数を扱う処理の一例を示すフローチャート

【図１０】図９に示された処理をを同プロセッサに行なわせるプログラムの例を示す図

【図１１】定数を格納するオペレーションと定数レジスタ３６の格納値を利用するオペレーションが分離しており、定数レジスタを明示的に指定してオペレーションを実行するプロセッサにおいて、図１０と同一の処理を行なわせるプログラムの例を示す図

【図１２】分岐オペレーションを含む処理を同プロセッサに行なわせるプログラムの例を示す図

【図１３】定数レジスタ３６の下位４ビットのみを変更するオペレーション“ｓｅｔ１４”を含む処理を同プロセッサに行なわせるプログラムの例を示す図

【図１４】定数レジスタの変形例に係る定数レジスタ３

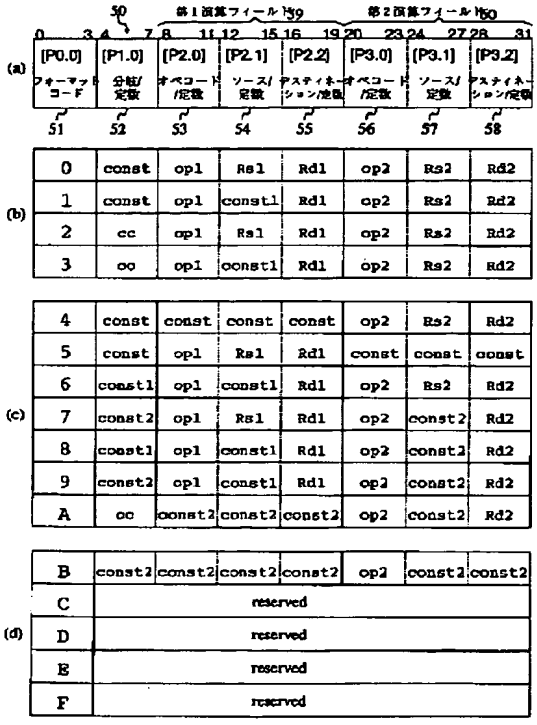
６の周辺回路の構成を示すブロック図

【符号の説明】

１０　命令レジスタ
２０　解読部
２１　フォーマットデコーダ
２２　命令デコーダ
２３　分岐デコーダ
２４　第１演算デコーダ
２５　第２演算デコーダ
２８　読み出しフラグ記憶部
３０　実行部
３１　実行制御部
３２　定数レジスタ制御部
３２ａ～３２ｈ　セレクタ
３３　ＰＣ部
３３ａ　固定値“４”
３３ｂ、３３ｅ　セレクタ

＊３３ｃ　加算器
３３ｄ　ＰＣ
３４　レジスタ群
３５　汎用レジスタＲ０～Ｒ１５
３６　定数レジスタＲ１６
３６ａ～３６ｈ　４ビット幅レジスタ
３７　第１演算部
３８　第２演算部
３９　命令フェッチ部
４０　オペランドアクセス部
５０　命令
５１～５８　命令フィールド
５９　第１演算フィールド
６０　第２演算フィールド
９０　変形例に係る定数レジスタ制御部
９１　格納桁カウンタ
＊

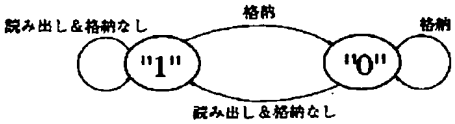
【図１】



【図２】

記号	オペレーション	ニーモニック表示
cc	分岐	eq,eqi,gt,gti,jmp,jmpi,...
op1	算術論理演算	add,sub,mul,and,or,cmp,...
	レジスタ間転送	mov,movh,movb,setl4
op2	算術論理演算	add,sub,mul,and,or,cmp,...
	レジスタ間転送	mov,movh,movb,setl4
	レジスタ・メモリ間転送	ld,ldh,ldb,st,sth,stb

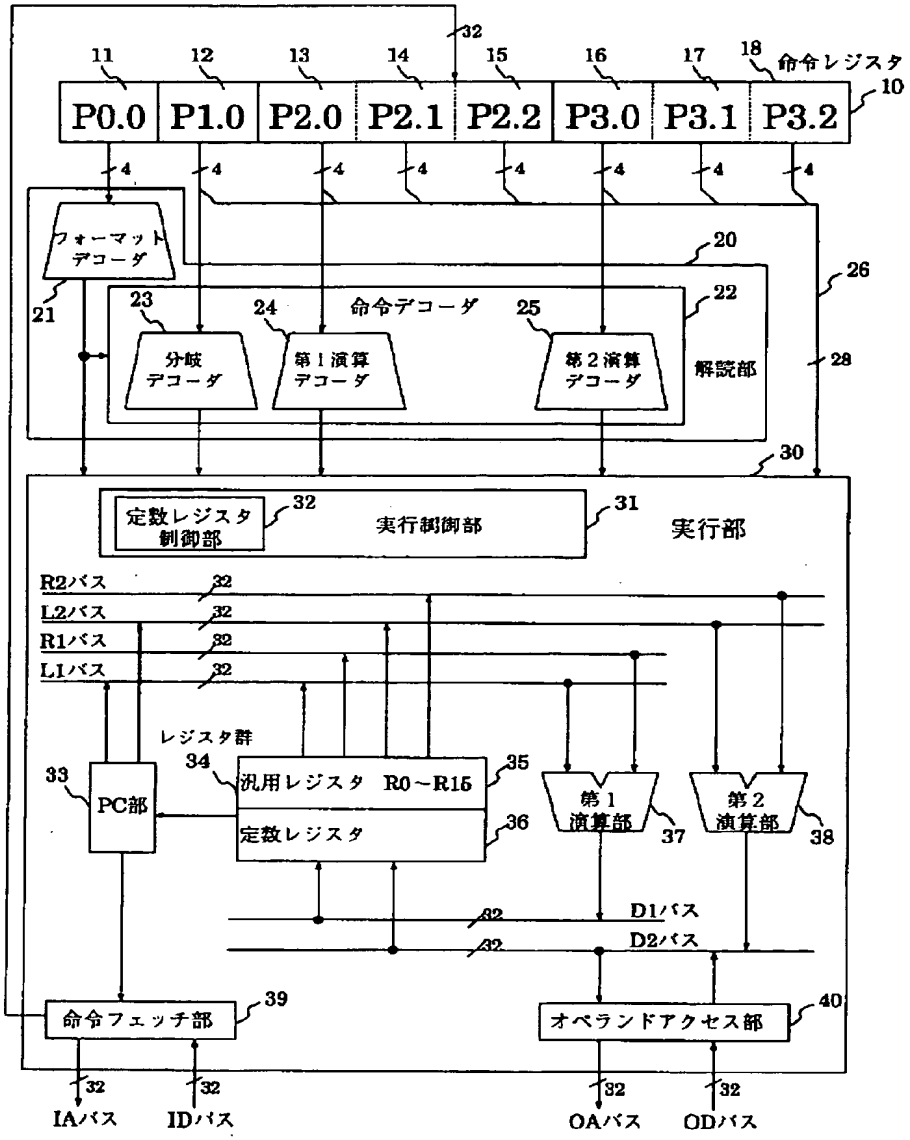
【図５】



【図７】

	36a	36b	36c	36d	36e	36f	36g	36h	28
(a)	8	7	6	5	4	3	2	1	1
(b)	0	0	0	0	0	0	0	8	0
(c)	0	0	0	0	0	0	8	7	0
(d)	0	0	0	0	0	8	7	6	0
(e)	0	0	0	0	8	7	6	5	0
(f)	0	0	0	0	8	7	6	5	1

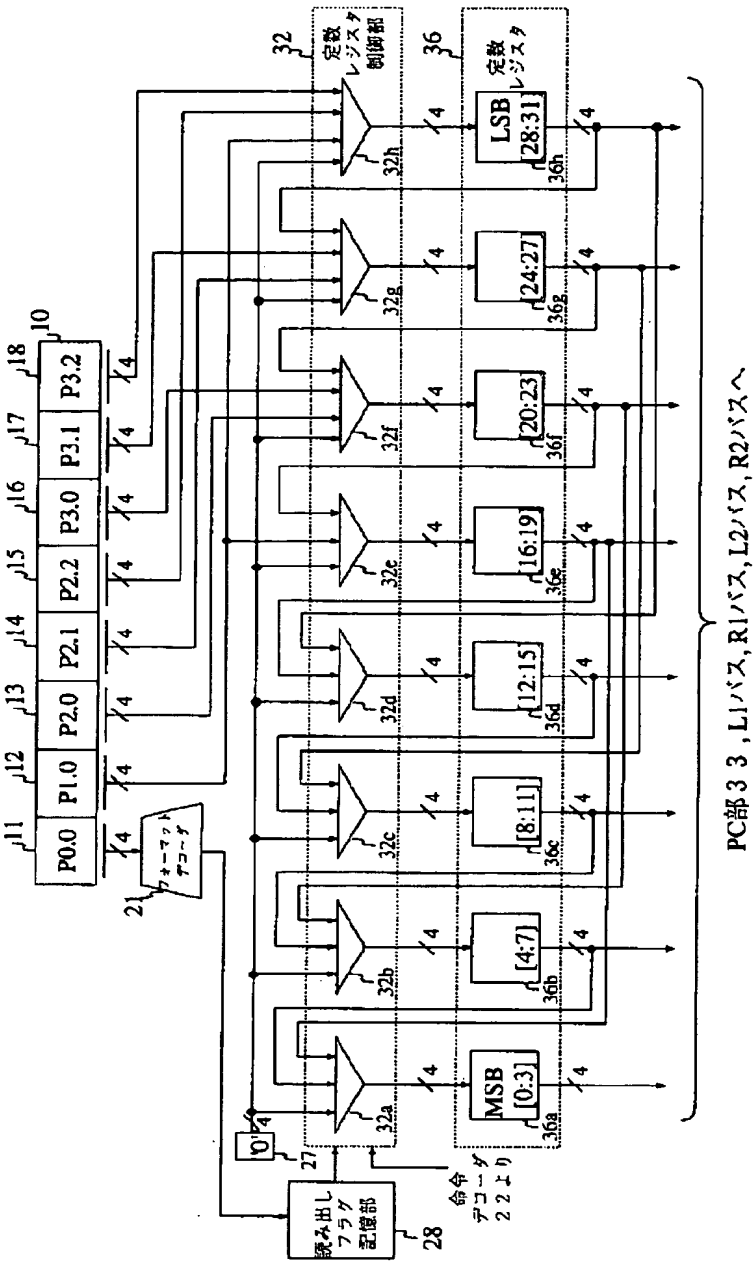
【図３】



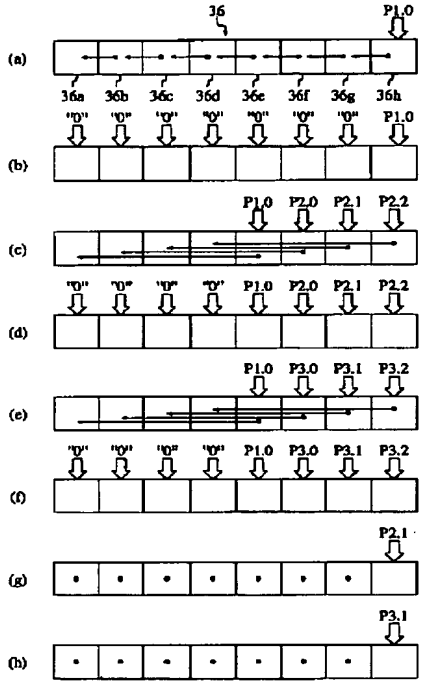
【図１０】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 4	0x8765				sub	R1	R0	71
fmt 6	0x4	add	0x3	R0	mov	R2	R1	72

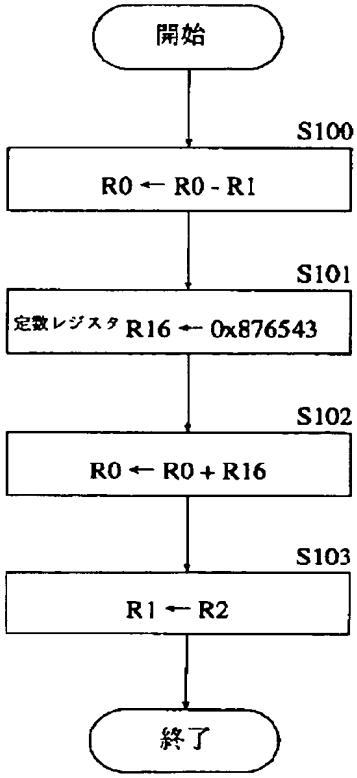
【図４】


P0.0
P1.0
P2.0
P2.1
P2.2
P3.0
P3.1
P3.2
デコード
定数レジスタ制御部
定数レジスタ
LSB [28:31]
[24:27]
[20:23]
[16:19]
[12:15]
[8:11]
[4:7]
MSB [0:3]
'0'
読み出しフラグ記憶部
命令デコーダ22より
PC部33, L1バス, R1バス, L2バス, R2バスへ

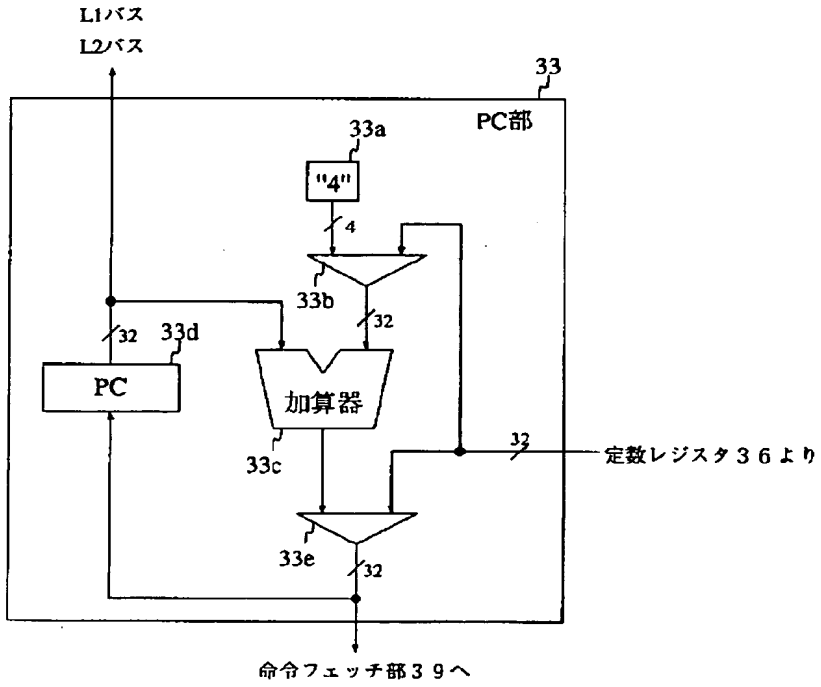
【図６】


36
P1.0
(a)
36a 36b 36c 36d 36e 36f 36g 36h
"0" "0" "0" "0" "0" "0" "0" P1.0
(b)
P1.0 P2.0 P2.1 P2.2
(c)
"0" "0" "0" "0" P1.0 P2.0 P2.1 P2.2
(d)
P1.0 P3.0 P3.1 P3.2
(e)
"0" "0" "0" "0" P1.0 P3.0 P3.1 P3.2
(f)
P2.1
(g)
P3.1
(h)

【図９】


開始
S100
R0 ← R0 - R1
S101
定数レジスタ R16 ← 0x876543
S102
R0 ← R0 + R16
S103
R1 ← R2
終了

【図８】


L1バス
L2バス
33
PC部
33a
"4"
4
33b
32
32
33d
PC
加算器
33c
32
定数レジスタ３６より
33e
32
命令フェッチ部３９へ

【図１１】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 4	0x0087				sub	R1	R0	73
fmt 4	0x6543				mov	R2	R1	74
fmt 2	nop	add	R15	R0	nop			75

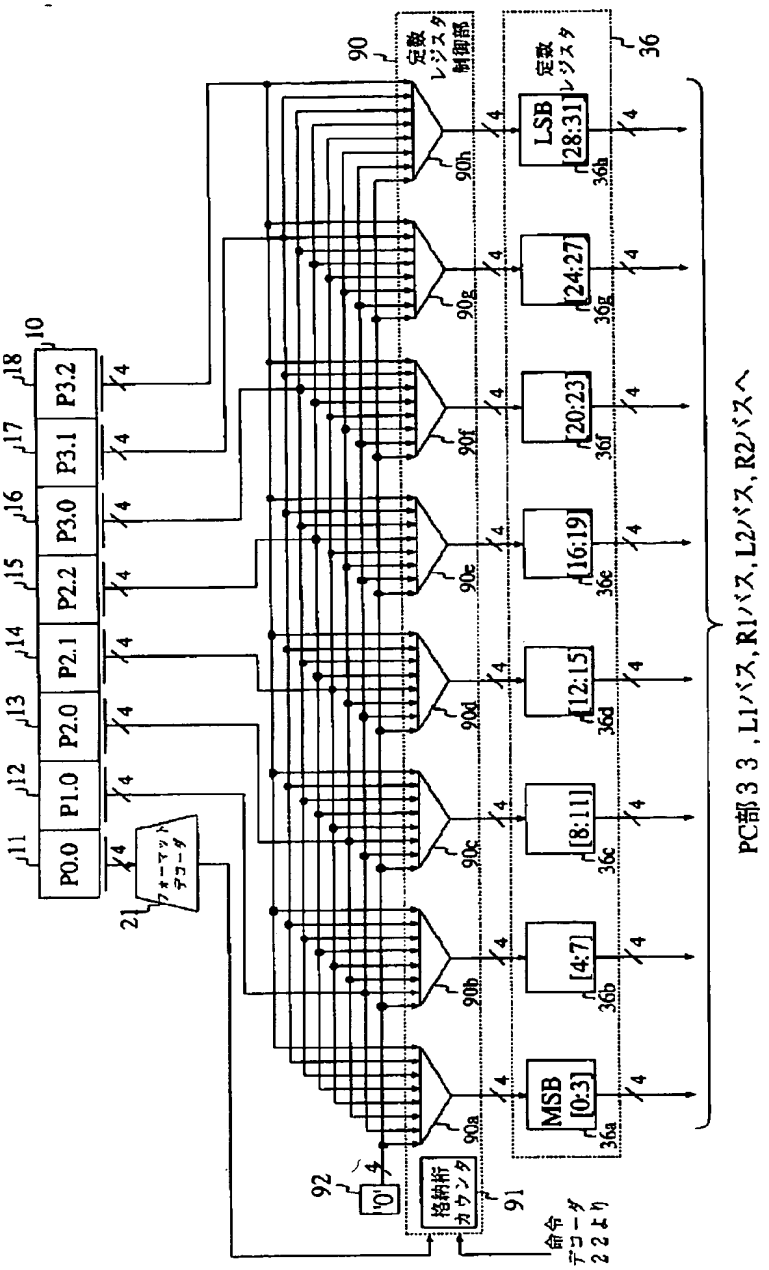
【図１２】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 4	0x1234				cmp	R0	R2	76
fmt 2	eqi	add	R2	R1	cmp	R4	R1	77
fmt 2	gti	add	R3	R2	mov	R2	R0	78

【図１３】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 4	0x3210				cmp	R0	R2	79
fmt 2	eqi	cmp	R2	R1	set14	0xc	—	80
fmt 2	gti	add	R3	R2	mov	R2	R0	81

【図１４】



フロントページの続き

(72)発明者　田中　哲也
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　高山　秀一
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　小谷　謙介
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　宮地　信哉
大阪府門真市大字門真1006番地　松下電器産業株式会社内